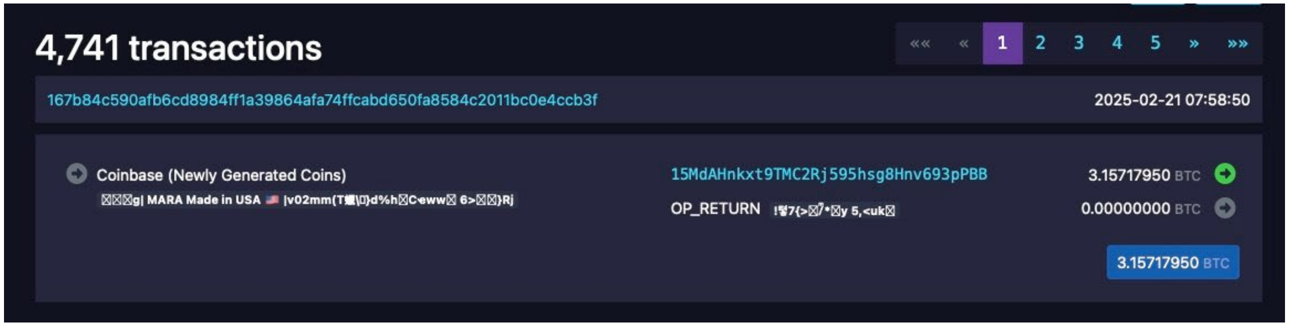
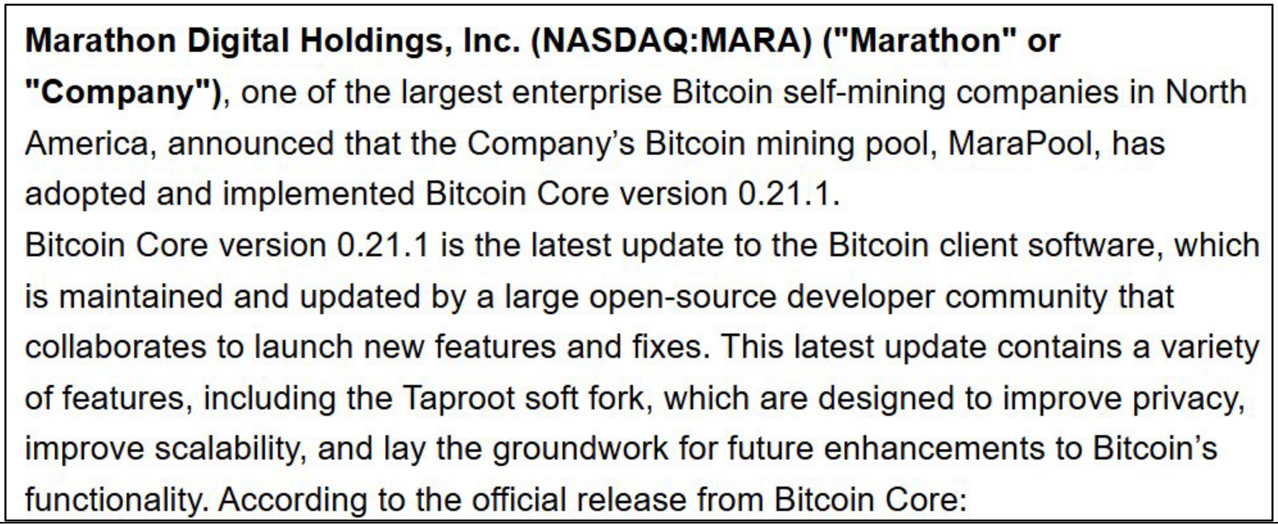
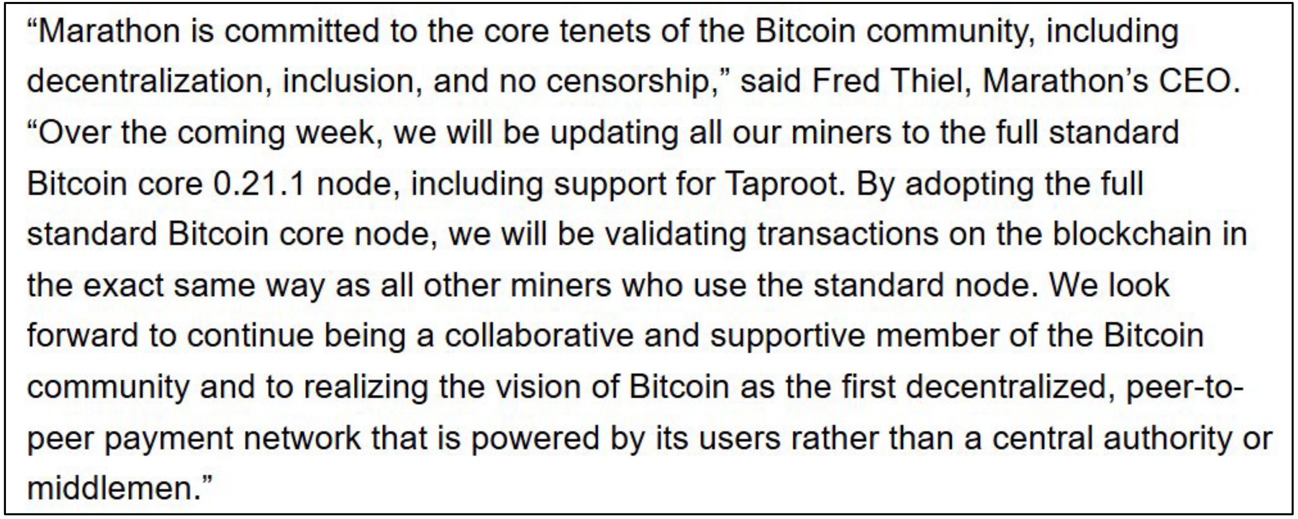
Exhibit 12

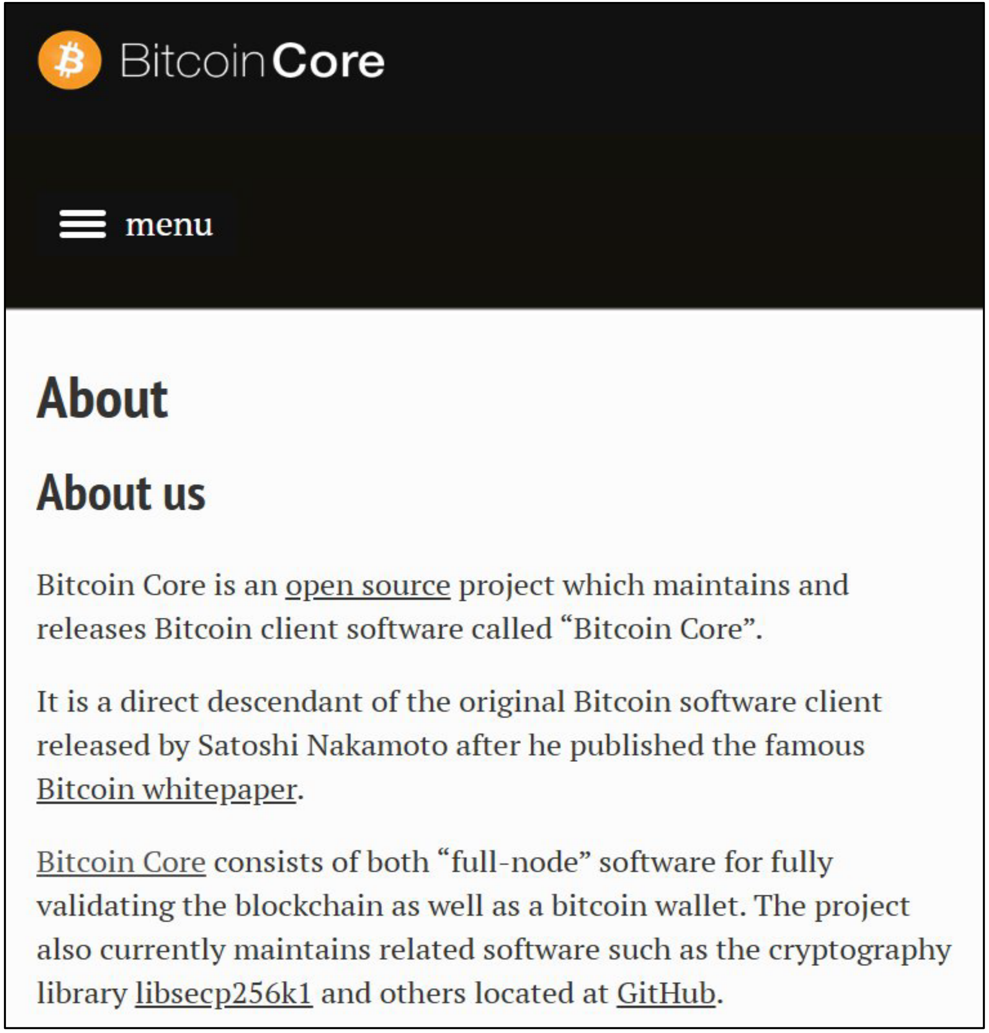
Exhibit 12: U.S. Patent No. 8,532,286

Claim 1	Exemplary Evidence of Infringement
[1pre] A method for performing, on a cryptographic apparatus, a Montgomery-style reduction in a cryptographic operation, the method comprising:	MARA Holdings, Inc. (hereinafter "MARA") performs, on a cryptographic apparatus (*e.g.*, a processor), a Montgomery-style reduction in a cryptographic operation (*e.g.*, during the transfer of Bitcoin to an address). *See*, *e.g.*: "Marathon is a digital asset technology company that is principally engaged in producing or **'mining' digital assets with a focus on the Bitcoin ecosystem** … **The term 'Bitcoin' with a capital 'B' is used to denote the Bitcoin protocol** which implements a highly available, public, permanent, and decentralized ledger." (Emphasis added) *See*, *e.g.*, MARA Holdings, Inc., Annual report pursuant to Section 13 and 15(d), (Form 10-K/A), at F-9, filed May 24, 2024, available at https://ir.mara.com/sec-filings/all-sec-filings/content/0001628280-24-025261/mara-20231231.htm. "The Bitcoin protocol is the technology that enables Bitcoin to function as a decentralized, peer-to-peer payment network. This open-source software, which sets the rules and processes that govern the Bitcoin network, is maintained and improved by a community of developers around the world known as Bitcoin Core developers … 'At Marathon, we have historically focused on supporting Bitcoin by adding hash rate, which helps secure the network, and now, we are supporting those who maintain **the open-source protocol on which we all depend** by contributing to Brink,' said Fred Thiel, Marathon's chairman and CEO." (Emphasis added) *See*, *e.g.*, Marathon Holdings Collaborates with Brink To Raise Up to $1 Million To Support Bitcoin Core Developers, GlobeNewswire (May 18, 2023), available at https://www.globenewswire.com/news-release/2023/05/18/2672276/0/en/Marathon-Digital-Holdings-Collaborates-with-Brink-To-Raise-Up-to-1-Million-To-Support-Bitcoin-Core-Developers.html. "The MaraPool wallet (Owned by the Company as Operator) is recorded on the distributed ledger as the winner of proof-of-work block rewards and assignee of all validations and, therefore, the transaction verifier of record. The pool participants entered into contracts with the Company as Operator; they did not directly enter into contracts with the network or the requester and were not

Claim 1	Exemplary Evidence of Infringement
	known verifiers of the transactions assigned to the pool…Therefore, the Company determined that it controlled the service of providing transaction verification services to the network and requester. **<u>Accordingly, the Company recorded all of the transaction fees and block rewards earned from transactions assigned to the MaraPool as revenue, and the portion of the transaction fees and block rewards remitted to the MaraPool participants as cost of revenues</u>**." (Emphasis added). *See*, *e.g.*, MARA Holdings., Inc., Quarterly report, (Form 10-Q), at Note 4 – Revenues, filed November 12, 2024, available at https://www.sec.gov/ix?doc=/Archives/edgar/data/0001507605/000162828024047148/mara-20240930.htm.  4,741 transactions 167b84c590afb6cd8984ff1a39864afa74ffcabd650fa8584c2011bc0e4ccb3f 2025-02-21 07:58:50 Coinbase (Newly Generated Coins) ⊠⊠⊠g| MARA Made in USA 🇺🇸 |v02mm(T⊠\⊠}d%h⊠Ceww⊠ 6>⊠⊠}Rj 15MdAHnkxt9TMC2Rj595hsg8Hnv693pPBB 3.15717950 BTC OP_RETURN !⊠7{>⊠⁷•⊠y 5,<uk⊠ 0.00000000 BTC 3.15717950 BTC *See*, *e.g.*, https://mempool.space/address/15MdAHnkxt9TMC2Rj595hsg8Hnv693pPBB. For example, MARA performs, on a cryptographic apparatus (*e.g.*, a processor), a Montgomery-style reduction in a cryptographic operation (*e.g.*, during the transfer of Bitcoin to an address) using the ECDSA signature protocol in the Bitcoin Core. "**<u>Bitcoin signed messages have three parts, which are the Message, Address, and Signature</u>**. The message is the actual message text - all kinds of text is supported, but it is recommended to avoid using non-ASCII characters in the signature because they might be encoded in different character sets, preventing signature verification from succeeding. The address is a legacy, nested segwit, or native segwit address. Message signing from legacy addresses was added by Satoshi himself and therefore does not have a BIP. **<u>Message signing from</u>**

Claim 1	Exemplary Evidence of Infringement
	segwit addresses has been added by BIP137 … The Signature is a base64-encoded ECDSA signature that, when decoded, with fields described in the next section.” (Emphasis added) *See*, *e.g.,* Message Signing, https://en.bitcoin.it/wiki/Message_signing. “This document describes a signature format for **signing messages with Bitcoin private keys**. The specification is intended to describe the standard for signatures of messages that can be signed and verified between different clients that exist in the field today.” (Emphasis added) *See*, *e.g.*, Bitcoin BIP137, https://github.com/bitcoin/bips/blob/master/bip-0137.mediawiki. For example, MARA uses Bitcoin Core, which comprises a method of performing, on a cryptographic apparatus (*e.g.*, a processor), a Montgomery-style reduction in a cryptographic operation (*e.g.*, during the transfer of Bitcoin to an address). *See*, *e.g.*:  **Marathon Digital Holdings, Inc. (NASDAQ:MARA) ("Marathon" or "Company")**, one of the largest enterprise Bitcoin self-mining companies in North America, announced that the Company’s Bitcoin mining pool, MaraPool, has adopted and implemented Bitcoin Core version 0.21.1. Bitcoin Core version 0.21.1 is the latest update to the Bitcoin client software, which is maintained and updated by a large open-source developer community that collaborates to launch new features and fixes. This latest update contains a variety of features, including the Taproot soft fork, which are designed to improve privacy, improve scalability, and lay the groundwork for future enhancements to Bitcoin’s functionality. According to the official release from Bitcoin Core:

Claim 1	Exemplary Evidence of Infringement
	 “Marathon is committed to the core tenets of the Bitcoin community, including decentralization, inclusion, and no censorship,” said Fred Thiel, Marathon’s CEO. “Over the coming week, we will be updating all our miners to the full standard Bitcoin core 0.21.1 node, including support for Taproot. By adopting the full standard Bitcoin core node, we will be validating transactions on the blockchain in the exact same way as all other miners who use the standard node. We look forward to continue being a collaborative and supportive member of the Bitcoin community and to realizing the vision of Bitcoin as the first decentralized, peer-to-peer payment network that is powered by its users rather than a central authority or middlemen.” *See*, *e.g.*, https://br.advfn.com/bolsa-de-valores/nasdaq/MARA/share-news/85244958/marathon-signals-for-taproot.

Claim 1	Exemplary Evidence of Infringement
	 BitcoinCore menu About About us Bitcoin Core is an open source project which maintains and releases Bitcoin client software called "Bitcoin Core". It is a direct descendant of the original Bitcoin software client released by Satoshi Nakamoto after he published the famous Bitcoin whitepaper. Bitcoin Core consists of both "full-node" software for fully validating the blockchain as well as a bitcoin wallet. The project also currently maintains related software such as the cryptography library libsecp256k1 and others located at GitHub. *See*, *e.g.*, Bitcoin core, https://github.com/bitcoin/bitcoin

Claim 1	Exemplary Evidence of Infringement

For example, during signature generation and verification a cryptographic operation (*e.g.*, `secp256k1_scalar_mul`) is invoked by the cryptographic apparatus (*e.g.*, a processor). *See*, *e.g.*:

```
static int secp256k1_ecdsa_sig_recover(const secp256k1_scalar *sigr, constsecp256k1_scalar* sigs,
  secp256k1_ge *pubkey, const secp256k1_scalar *message, int recid)
{
    ...;
    secp256k1_scalar_mul(&u2, &rn, sigs);
    ...;
secp256k1_scalar_mul(&u2, &rn, sigs);
...;
}
```

See, *e.g.*, src/secp256k1/src/modules/recovery/main_impl.h

```
static int secp256k1_ecdsa_sig_sign(const secp256k1_ecmult_gen_context *ctx, secp256k1_scalar *sigr,
  secp256k1_scalar *sigs, const secp256k1_scalar *seckey, const secp256k1_scalar *message,
  const secp256k1_scalar *nonce, int *recid)
{
    ...;
    secp256k1_scalar_mul(&n, sigr, seckey);
    ...;
    secp256k1_scalar_mul(sigs, sigs, &n);
    ...;
}
```

See, *e.g.*, src/secp256k1/src/ecdsa_impl.h

MARA performs the method using a cryptographic apparatus. *See*, *e.g.*:

"Our core business is bitcoin mining, and we produce, or 'mine,' bitcoin using one of the industry's largest and most energy-efficient fleets of **specialized computers** while providing dispatchable compute as an optionality to the electric grid operators to balance electric demands on the grid." (Emphasis added)

See, *e.g.*, MARA Holdings, Inc., Form 10-K/A, at 6, filed March 3, 2025, available at https://ir.mara.com/sec-filings/all-sec-filings/content/0001628280-24-025261/mara-20231231.htm.

Claim 1	Exemplary Evidence of Infringement
	"Over the past three years, digital asset mining operations have evolved from individual users mining with **computer processors, graphics processing units and first-generation mining rigs**. New processing power brought onto the digital asset networks is predominantly added by professionalized mining operations, which may use **proprietary hardware or sophisticated machines**." (Emphasis added) *See*, *e.g.*, MARA Holdings, Inc., Form 10-K/A, at 21, filed March 3, 2025, available at https://ir.mara.com/sec-filings/all-sec-filings/content/0001628280-24-025261/mara-20231231.htm. "As of December 31, 2024, we operated approximately 400,000 bitcoin mining **ASICs**, capable of producing 53.2 EH/s with an efficiency of 19.2 joules per terahash, which is among the most efficient in the industry." (Emphasis added) *See*, *e.g.*, MARA Holdings, Inc., Form 10-K/A, at 21, filed March 3, 2025, available at https://ir.mara.com/sec-filings/all-sec-filings/content/0001628280-24-025261/mara-20231231.htm. "Miners, which operate **specialized hardware, known as bitcoin mining rigs or application-specific integrated circuits ("ASICs")**, then compete to process these unconfirmed transactions into a 'block.'" (Emphasis added) *See*, *e.g.*, MARA Holdings, Inc., Form 10-K/A, at 6, filed March 3, 2025, available at https://ir.mara.com/sec-filings/all-sec-filings/content/0001628280-24-025261/mara-20231231.htm.
[1a] obtaining an operand for the cryptographic operation;	MARA obtains an operand for the cryptographic operation; computes a modified operand using a reduction value, instead of a modulus used in performing a standard Montgomery reduction, to perform a replacement of a least significant word of the operand, rather than perform a cancellation thereof, the reduction value being a function of the modulus; and outputs the modified operand. *See*, *e.g.*:

<table>
<tr><th>Claim 1</th><th>Exemplary Evidence of Infringement</th></tr>
<tr><td>

[1b] computing a modified operand using a reduction value, instead of a modulus used in performing a standard Montgomery reduction, to perform a replacement of a least significant word of the operand, rather than perform a cancellation thereof, the reduction value being a function of the modulus; and

[1c] outputting the modified operand.

</td><td>

For example, operand `l` is 8 machine words, reduced operand `r` is 4 machine words. *See*, *e.g.*:

```
/** Set a scalar to an unsigned integer. */
static void secp256k1_scalar_set_int(secp256k1_scalar *r, unsigned int v);
```

See, *e.g.*, src/secp256k1/src/scalar.h

```
/* Limbs of the secp256k1 order. */
#define SECP256K1_N_0 ((uint64_t)0xBFD25E8CD0364141ULL)
#define SECP256K1_N_1 ((uint64_t)0xBAAEDCE6AF48A03BULL)
#define SECP256K1_N_2 ((uint64_t)0xFFFFFFFFFFFFFFFEULL)
#define SECP256K1_N_3 ((uint64_t)0xFFFFFFFFFFFFFFFFULL)

/* Limbs of 2^256 minus the secp256k1 order. */
#define SECP256K1_N_C_0 (~SECP256K1_N_0 + 1)
#define SECP256K1_N_C_1 (~SECP256K1_N_1)
#define SECP256K1_N_C_2 (1)

SECP256K1_INLINE static void secp256k1_scalar_set_int(secp256k1_scalar *r, unsigned int
v) {
    r->d[0] = v;
    r->d[1] = 0;
    r->d[2] = 0;
    r->d[3] = 0;
    ...;
}

static void secp256k1_scalar_mul(secp256k1_scalar *r, const secp256k1_scalar *a, const
secp256k1_scalar *b) {
    uint64_t l[8];
    ...;
    secp256k1_scalar_mul_512(l, a, b);
    secp256k1_scalar_reduce_512(r, l);
    ...;
}
```

See, *e.g.*, src/secp256k1/src/scalar_4x64_impl.h (*see also* code in “scalar_8x32_impl.h”)

</td></tr>
</table>

<table>
<tr><th>Claim 1</th><th>Exemplary Evidence of Infringement</th></tr>
<tr><td></td><td>

For example, operand `l` is reduced until it fits in p[0..4], which is further reduced into `r`. *See*, *e.g.*:

```
SECP256K1_INLINE static int secp256k1_scalar_reduce_512(secp256k1_scalar *r, const
uint64_t *l) {
    secp256k1_uint128 c128;
    ...;
    uint64_t n0 = l[4], n1 = l[5], n2 = l[6], n3 = l[7];
    ...;
    /* Reduce 512 bits into 385. */
    /* m[0..6] = l[0..3] + n[0..3] * SECP256K1_N_C. */
    ...;
    /* Reduce 385 bits into 258. */
    /* p[0..4] = m[0..3] + m[4..6] * SECP256K1_N_C. */
    ...;
    /* Reduce 258 bits into 256. */
    /* r[0..3] = p[0..3] + p[4] * SECP256K1_N_C. */
    secp256k1_u128_from_u64(&c128, p0);
    secp256k1_u128_accum_mul(&c128, SECP256K1_N_C_0, p4);
    r->d[0] = secp256k1_u128_to_u64(&c128); ...;
}
```

See, *e.g.*, src/secp256k1/src/scalar_4x64_impl.h (*see also* code in “scalar_8x32_impl.h”)

</td></tr>
</table>